The Department of Homeland Security (DHS) Federal Emergency Management Agency (FEMA)/ National Training and Education Division (NTED)

Registration and Evaluation System (RES) Hands-On XML training

December 20, 2010

FEMA

Overview of the RES

NEED

A comprehensive, centralized database of training participant numbers and training evaluations

Enhances ability to manage the quality of their sponsored courses and provide performance reporting

SOLUTION

A dedicated system and accompanying processes

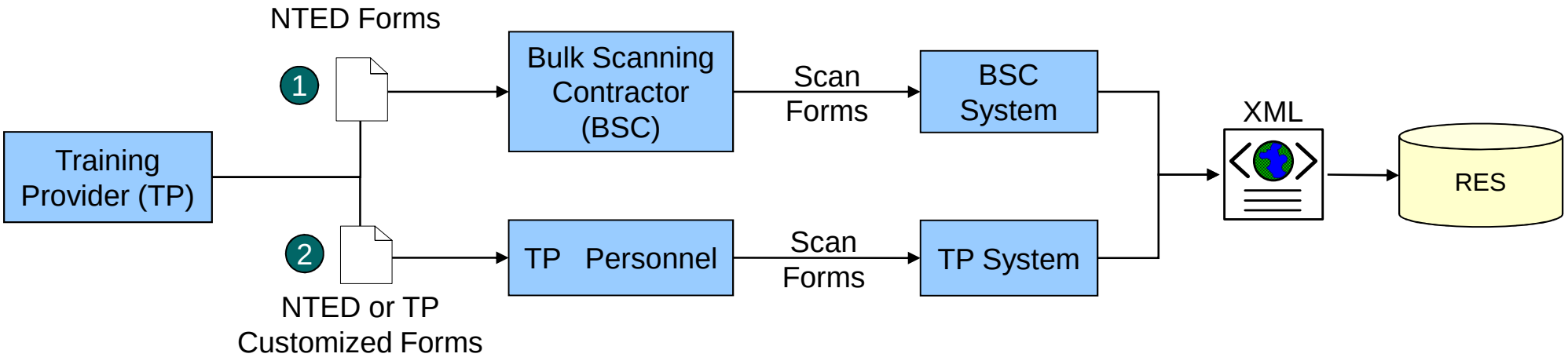
- Collection of registration information (to collect training numbers)
- Level One: Post course evaluation

Level Two: Pre-and-post performance test

- Established standard processes and operating procedures for submitting course registration and evaluation data

FEMA

Key to Success: Process of Data Collection and Submission



1 Scenario 1: The TP will be responsible for sending the forms to the BSC. The BSC will scan the forms, output the XML file, and upload the XML to the RES

2 Scenario 2: The TP will be responsible for scanning the forms, outputting the XML file, and uploading the XML to the RES

FEMA

Training Session Objectives

Each TP should be able to:

- ❑ Understand the concepts of an XML
- ❑ Write an XML
- ❑ Understand the concepts of a DTD and interpret its syntax
- ❑ Write an XML based on specified DTD
- ❑ Understand the basic concepts of data mapping
- ❑ Name an XML file based on the RES File Name Convention
- ❑ Write an XML based on specified RES DTD
- ❑ Submit an XML file via the RES
- ❑ Interpret an error log (if applicable)
- ❑ Re-name an XML file and re-submit via the RES (if applicable)

Overview: What is XML?

XML is defined as an e**X**tensible **M**arkup **L**anguage (XML)

– Its primary purpose is to facilitate the sharing of structured data across different information systems

XML is a generic framework to store data in a tree structure that allows the technical team to define the XML tags

– XML is created to structure, store and transfer data

Example:

```
<student>
        <lastname>Doe</lastname>
        <firstname>John</firstname>
        <phone>1234567890</phone>
</student>
```

FEMA

XML is Just Plain Text

XML is a computer language used for transferring data that is software and hardware independent

- Developers can write a program to:
  - Create or read XML
  - To store or send data
  - To receive and display data

If the software can handle plain text, then it will also be able to handle XML

Users can define and create their own structure and tags

- The tags <student>, <lastname> from the previous example are not defined in any XML standards but defined by the user

FEMA

XML Tree

The XML filename has an extension of .xml (i.e., submission.xml)

The XML document forms a tree structure that starts at the **root** element and branches to the **children** elements

```
<student>
        <lastname>Doe</lastname>
        <firstname>John</firstname>
        <phone>1234567890</phone>
</student>
```

<student> is the root element that has 3 child elements <lastname>, <firstname>, and <phone>

FEMA

XML Structure

The first line of an XML document is the **declaration** of the XML version and mark-up language

<?xml version="1.0" encoding="UTF-8" ?>

The next lines after the declaration contain the contents of the file, which are defined by the root element and its child elements

All XML elements must have opening tags **(<name>)** and closing tags **(</name>)**

– XML tags are case-sensitive. All opening tags and closing tags must be written in the same case format

<firstname>Joe</firstname> valid syntax

<Firstname>Joe</firstname> invalid syntax

FEMA

XML Elements

Each XML element contains the contents from opening tag to closing tag
<name>…**</name>**

XML elements must follow these naming rules:

- It can contain letters, numbers, and other characters
- It cannot begin with a number, punctuation character or contain spaces

Best Practices in Element Naming Convention:

- It should be short and simple: <firstname>
- Avoid hyphens (-), colons, or periods: <first-name>, <first.name>

An XML element can have child elements or values

```
<student>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
</student>
```

FEMA

XML Attributes

Attributes provide additional information about elements

- It must follow the same naming rules as elements
- It must be in the opening tag of elements

Attribute values must be enclosed in quotes and must be in the same format.

<student name="John Doe"> </student> valid

NOTE: Please use double quotes when submitting data to the RES because the XML file will be loaded into an Oracle database and may cause an error if it contains single quotes

XML Elements vs. Attributes: Various ways to represent data

There are no rules of when to use elements and when to use attributes

Sample of data using **elements**

```
<student>
        <lastname>Doe</lastname>
        <firstname>John</firstname>
</student>
```

Sample of data using **attributes**

```
<student lastname="Doe" firstname="John"></student>
```

When a validation rule document (such as a Document Type Definition (DTD)) is used, elements must be in the order that is defined in the validation document. However, attributes do not need to follow the order.

FEMA

XML Validation

A valid XML has to be “well-formed” and pass all validation rules

A “Well-Formed” XML is an XML document that has valid XML syntax

- It must have a root element
- Its elements must have opening tags and closing tags
- XML tags are case sensitive
- XML elements must be properly nested

```
<student>
    <name></name>
</student>
```
valid

```
<student><name></student></name>
```
invalid

- XML attribute values must be quoted

The validation rules and business logic of a XML document is validated against a Document Type Definition (DTD) document using a program written by developers

FEMA

Training Session Objectives

Each TP should be able to:

- ☑ Understand the concepts of an XML
- ❑ Write an XML
- ❑ Understand the concepts of a DTD and interpret its syntax
- ❑ Write an XML based on specified DTD
- ❑ Understand the basic concepts of data mapping
- ❑ Name an XML file based on the RES File Name Convention
- ❑ Write an XML based on specified RES DTD
- ❑ Submit an XML file via the RES
- ❑ Interpret an error log (if applicable)
- ❑ Re-name an XML file and re-submit via the RES (if applicable)

Exercise 1: Writing an XML

Write an XML using the following data:

	Student 1	Student 2	Student 3
Last Name	Bardeen	Curie	Pauling
First Name	John	Marie	Linus
Middle Name		A	R
Email	Bardeen_J@hotmail.com		Linus.R.Pauling@gmail.com
City	Birmingham	Chapel Hill	Philadelphia
State	Al	nc	PA

The XML must have **class** as the root element and **student** as a child element of **class**

NOTE: There are many correct solutions to this exercise. Please make sure that the naming conventions in your solution are consistent throughout the XML

FEMA

Training Session Objectives

Each TP should be able to:

- ☑ Understand the concepts of an XML
- ☑ Write an XML
- ❑ Understand the concepts of a DTD and interpret its syntax
- ❑ Write an XML based on specified DTD
- ❑ Understand the basic concepts of data mapping
- ❑ Name an XML file based on the RES File Name Convention
- ❑ Write an XML based on specified RES DTD
- ❑ Submit an XML file via the RES
- ❑ Interpret an error log (if applicable)
- ❑ Re-name an XML file and re-submit via the RES (if applicable)

FEMA

Overview: What is a Document Type Definition (DTD)?

A DTD defines the tree structure, the legal elements and attributes, and the list of acceptable values for attributes of an XML

- With a DTD, a group of users can agree to a standard structure and format for exchanging their data
- Users can verify that data received from other users are valid prior to further process
- Users sending an XML file can verify that their data is valid prior to sending to other users

DTD file has an extension of .dtd (i.e., submission.**dtd**) and can be handled by any software application that can handle plain text

A DTD is used to define the structure of a XML as well as the definitions of:

- XML Elements
- XML Attributes
- CDATA – character data

FEMA

DTD Elements

In the DTD, XML elements are declared as < !ELEMENT>

To declare <student> element

– Syntax: <!ELEMENT student>

	To declare an EMPTY element with no value	To declare an element with list of character data
Syntax	<!ELEMENT student **EMPTY**>	<!ELEMENT state **(#CDATA|CA|FL)**>
Example in XML	<student></student>	<state>CA</state> OR <state>FL</state> valid <state>TX</state> invalid

FEMA

DTD Elements - Syntax

	To declare an element with child element	To declare an element with children (sequences)
Syntax	<!ELEMENT student **(name)**>	<!ELEMENT student **(name,phone)**>
Example in XML	<student> <name></name> </student>	<student> <name></name> <phone></phone> </student> valid <student> <phone></phone> <name></name> </student> invalid

***NOTE:** The <name> and <phone> elements must be in the sequential order as declared in the DTD for the XML to be valid

FEMA

DTD Elements – Occurrence Symbols

	To declare occurrences of an element	
	ONLY ONE	**Minimum ONE**
Syntax	<!ELEMENT student (name)>	<!ELEMENT company (location**+**)>

	To declare occurrences of an element	
	Zero or One	**Zero or More**
Syntax	<!ELEMENT student (fax*****)>	<!ELEMENT training (instructor**?**)>

FEMA

DTD Attributes

In the DTD, the XML Attribute is declared as < !ATTLIST>

	To declare an attribute, the tag has to include element name
Syntax	<!ATTLIST elementName attributeName>
Example in XML	<!ATTLIST student phonenumber> <student phonenumber="1234567890"></student>

DTD Attributes - Syntax

	To declare a REQUIRED attribute	To declare an IMPLIED attribute
Syntax	<!ATTLIST student phone CDATA **#REQUIRED**>	<!ATTLIST student phone CDATA **#IMPLIED**>
Example in XML	<student phone="1234567890"></student valid <student></student> invalid	<student></student> OR <student phone="1234567890"></student>

FEMA

DTD Attributes - Syntax

	To declare an ENUMERATED (list of values) attribute
Syntax	<!ATTLIST elementName attributeName (value1|value2)> Example: <!ATTLIST payment type (check|cash)>
Example in XML	<payment type="check"></payment> OR <payment type="cash"></payment>

FEMA

Link the XML and DTD

Syntax to include a DTD in a XML

– Written in the second line of the XML after the declaration:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE student SYSTEM “validate.dtd">
<student>
  ……….
</student>
```

Both the DTD and the XML files must be in the same directory

Training Session Objectives

Each TP should be able to:

- ☑ Understand the concepts of an XML
- ☑ Write an XML
- ☑ Understand the concepts of a DTD and interpret its syntax
- ❑ Write an XML based on specified DTD
- ❑ Understand the basic concepts of data mapping
- ❑ Name an XML file based on the RES File Name Convention
- ❑ Write an XML based on specified RES DTD
- ❑ Submit an XML file via the RES
- ❑ Interpret an error log (if applicable)
- ❑ Re-name an XML file and re-submit via the RES (if applicable)

FEMA

Exercise 2: Write an XML based on the DTD

Using the XML from Exercise 1, modify it based on the following DTD "**exercise2.dtd**":

```
<!ELEMENT class (student+)>
<!ELEMENT student EMPTY>
        <!ATTLIST student
        lastname CDATA #REQUIRED
        firstname CDATA #REQUIRED
        mi CDATA #IMPLIED
        state
        (AL|AK|AZ|AR|CA|CO|CT|DC|DE|FL|GA|HI|ID|IL|IN|IA|KS|KY|LA|ME|MD|MA|MI|MN|MS|MO|MT|NE|NV|NH|NJ|NM
        |NY|NC|ND|OH|OK|OR|PA|RI|SC|SD|TN|TX|UT|VT|VA|WA|WV|WI|WY|AS|FM|GU|MH|MP|PW|PR|VI|AA|AE|AP)        #REQUIRED
        email CDATA #IMPLIED>
```

Exercise 2 Solution

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE class SYSTEM "exercise2.dtd">

<class>
<student lastname="Bardeen" firstname="John" city="Birmingham" state="AL"
email="Bardeen_J@hotmail.com"></ student>

<student lastname="Curie" firstname="Marie" mi="A" city="Chapel Hill" state="NC"
email="MCurie@yahoo.com" ></student>

<student lastname="Pauling" firstname="Linus" mi="R" city="Philadelphia" state="PA"
email="Linus.R.Pauling@gmail.com"></student>
</class>
```

FEMA

NOTE: Attributes do not necessarily have to be in the order as indicated in the solution

Training Session Objectives

Each TP should be able to:

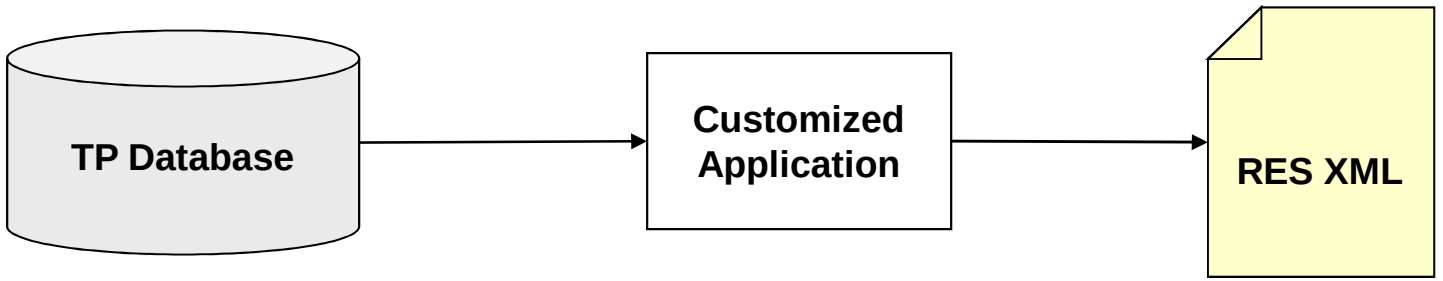
- ☑ Understand the concepts of an XML
- ☑ Write an XML
- ☑ Understand the concepts of a DTD and interpret its syntax
- ☑ Write an XML based on specified DTD
- ❑ Understand the basic concepts of data mapping
- ❑ Name an XML file based on the RES File Name Convention
- ❑ Write an XML based on specified RES DTD
- ❑ Submit an XML file via the RES
- ❑ Interpret an error log (if applicable)
- ❑ Re-name an XML file and re-submit via the RES (if applicable)

Overview: Mapping Data Fields to the Defined XML Tags

The goal behind data mapping is to map data elements from a database to the defined data fields in a DTD

The Training Provider (TP) will need to identify technical staff to:

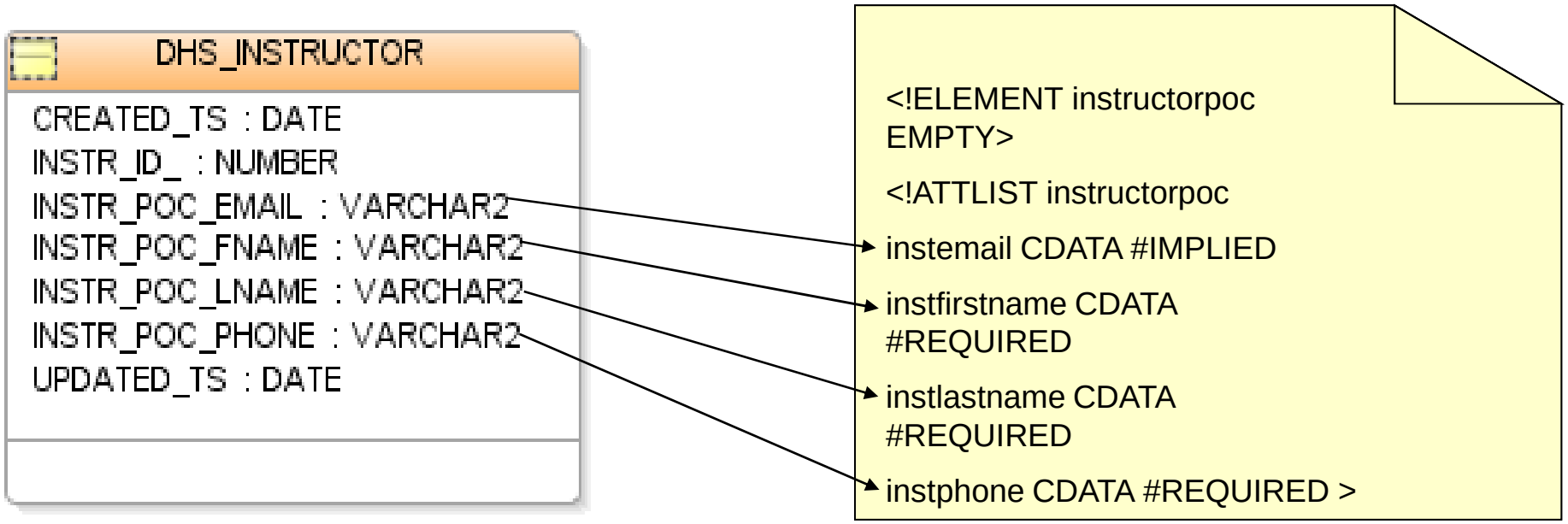
1. Map the database data elements to the defined data elements in the DTD
2. Develop a customized program to perform the export/conversion of data fields to the appropriate data elements



FEMA

Example of Mapping Tables to DTD Tags

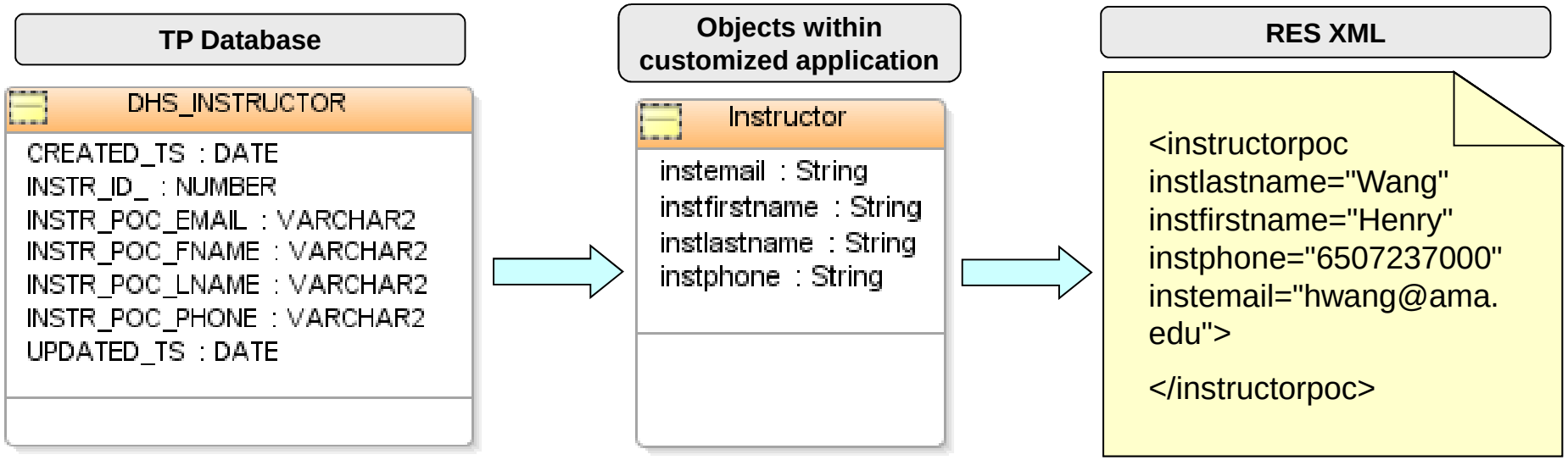
The structure of the document should be such that the column data can be represented by either elements or attributes.



FEMA

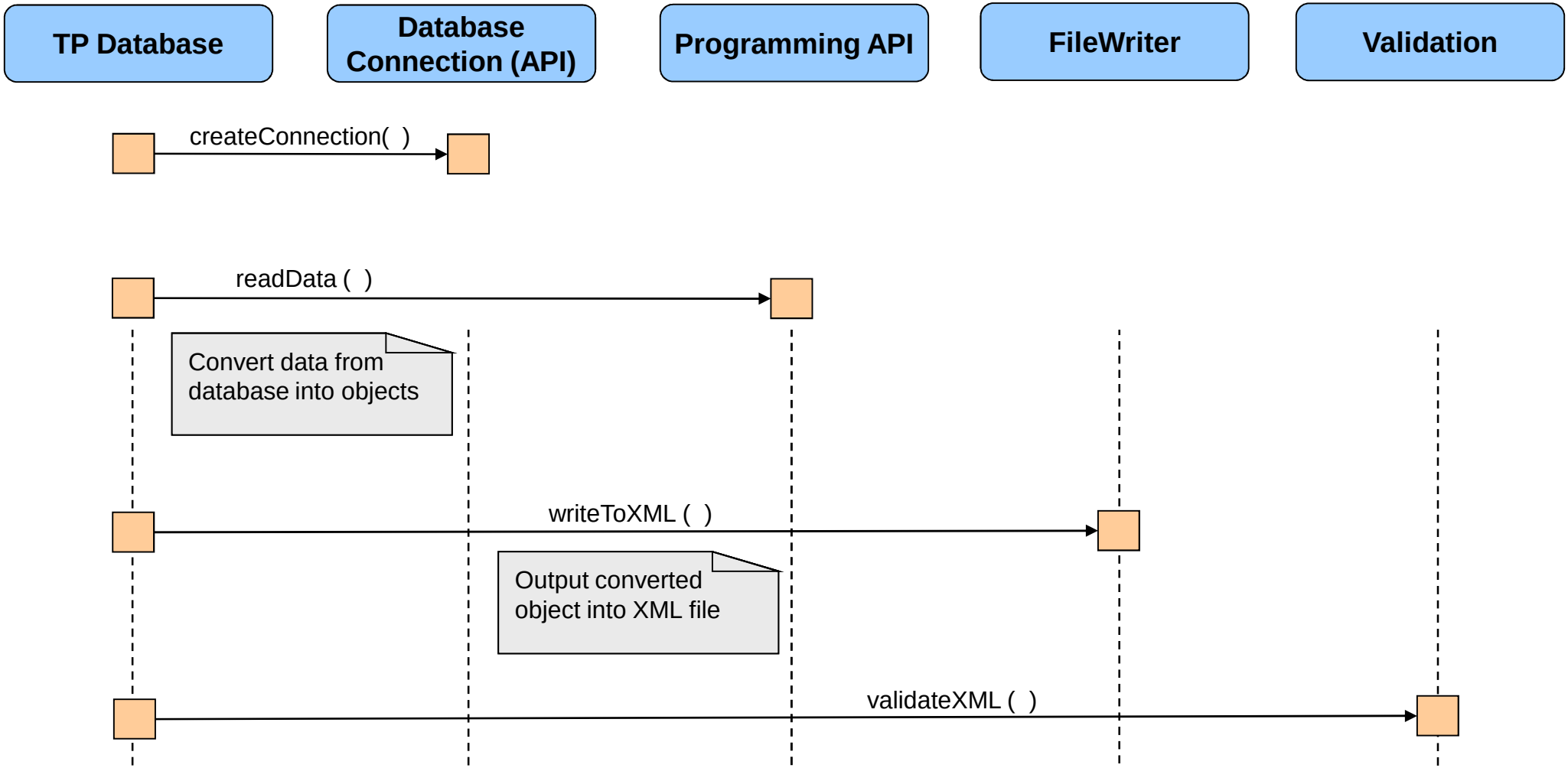
Converting Data Fields to Multiple Objects

Once the data elements are mapped to the defined DTD tags, technical staff will develop a customized program to convert the database columns into the defined DTD tags



FEMA

Example of Customized Application



FEMA

Training Session Objectives

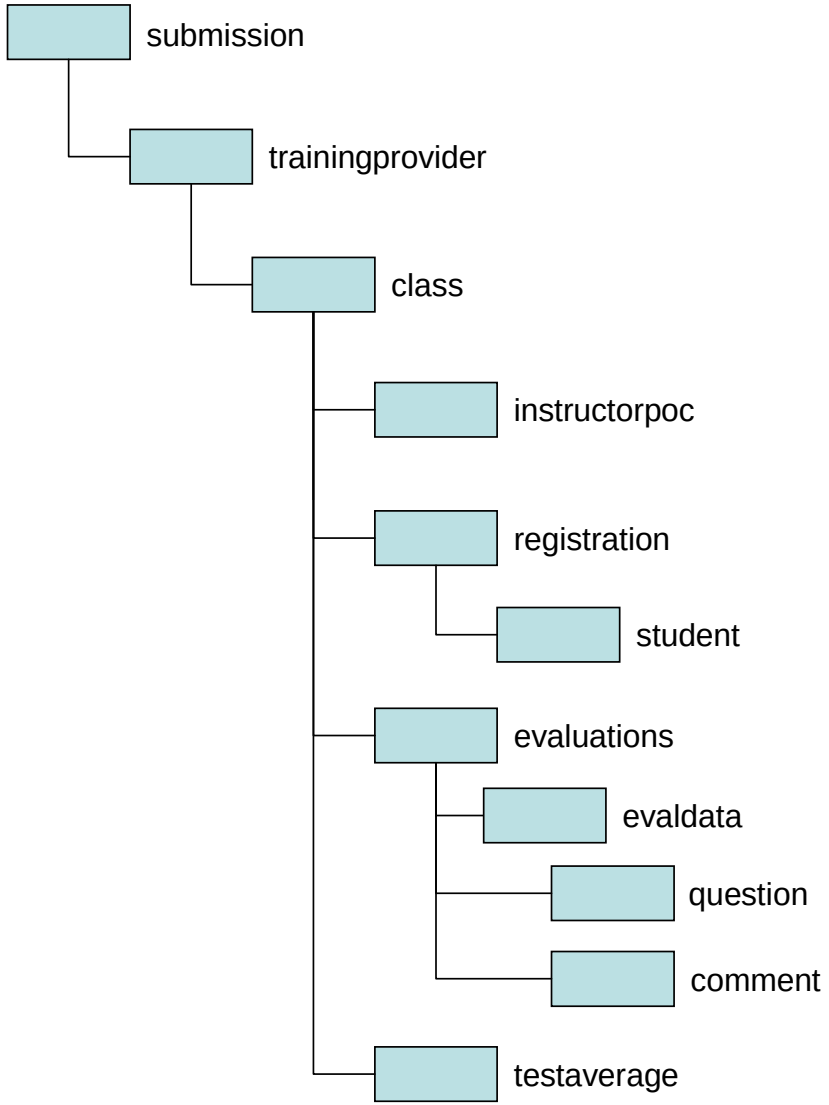
Each TP should be able to:

- ☑ Understand the concepts of an XML
- ☑ Write an XML
- ☑ Understand the concepts of a DTD and interpret its syntax
- ☑ Write an XML based on specified DTD
- ☑ Understand the basic concepts of data mapping
- ❑ Name an XML file based on the RES File Name Convention
- ❑ Write an XML based on specified RES DTD
- ❑ Submit an XML file via the RES
- ❑ Interpret an error log (if applicable)
- ❑ Re-name an XML file and re-submit via the RES (if applicable)

Structure of the RES DTD

The layout of the RES DTD is displayed

– Each box represents elements and/or sub-elements. Each element can consist of multiple attributes and child elements



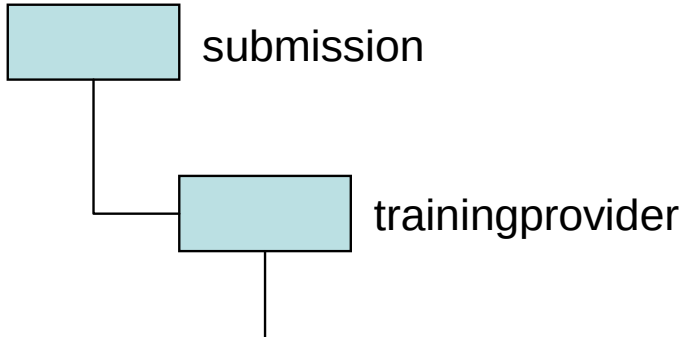
FEMA

Elements: submission and trainingprovider

The **submission** element does not have any attributes but has a child element, **trainingprovider**

The **trainingprovider** element provides the general information of a training provider (TP)

– It contains three attributes which are all **Required**
  - *tpid*
  - *tpphone*
  - *tpemail*



Indicates at a minimum one class information required

```
<!ELEMENT trainingprovider (class+)>
<!ATTLIST trainingprovider
        tpid (ACEP|AMA|APRI|ARC|ASU|BSU|CDP|CRD|COSA|…) #REQUIRED
        tpphone CDATA #REQUIRED
        tpemail CDATA #REQUIRED>
```

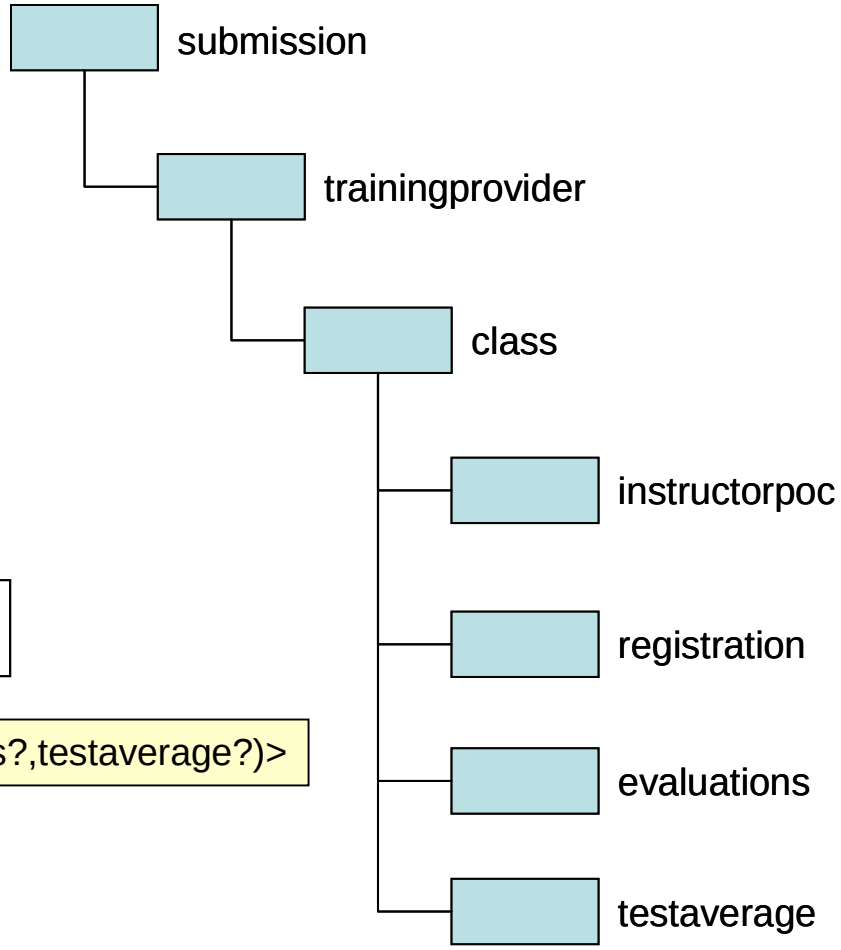
FEMA

Element: class

The **trainingprovider** element has a child element, **class**

The **class** element has four child elements

- **instructorpoc**
- **registration**
- **evaluations**
- **testaverage**



Indicates that zero or more than one occurrence

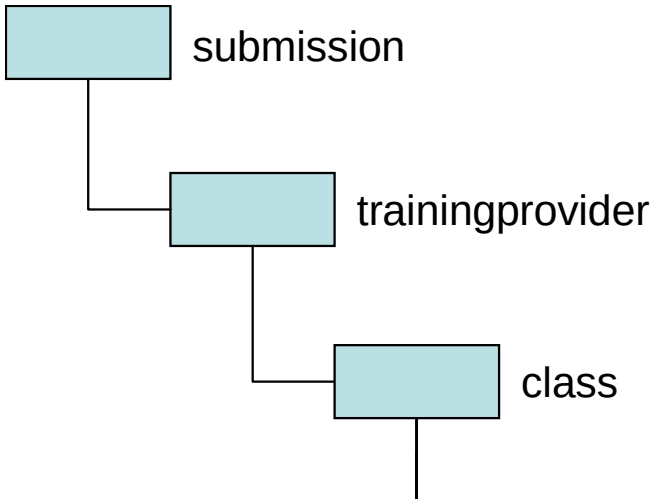
<!ELEMENT class (instructorpoc?,registration?,evaluations?,testaverage?)>

FEMA

Element: class

The **class** element also has multiple attributes

- It has a mix of required and implied attributes
- Pay particular attention to the attribute, *classtype*



There are three possible values for *classtype*

- Instructor-led (I)
- Web-based Domestic (WD)
- Web-based International (WI)

Depending on the chosen value for *classtype*, implied attributes and child elements may become required

FEMA

If the attribute *classtype* is "I" for Instructor-led, the following implied attributes will become required

<!ATTLIST class

classtype = "I"

catalognum (AWR-103|AWR-110-W|AWR-111-W|…) #REQUIRED

classcity CDATA #REQUIRED

classstate (AL|AK|AZ|AR|CA|CO|…) #IMPLIED

classzipcode CDATA #IMPLIED

classcountry (AA|AC|AE|AF|…..) #IMPLIED

startdate CDATA #REQUIRED

enddate CDATA #REQUIRED

starttime CDATA #IMPLIED

endtime CDATA #IMPLIED

numstudent CDATA #REQUIRED

trainingmethod (R|M|I|W) #REQUIRED

contacthours CDATA #IMPLIED

preparerlastname CDATA #IMPLIED

preparerfirstname CDATA #IMPLIED

batchpreparerphone CDATA #IMPLIED

batchpreparemail CDATA #IMPLIED>

Attributes that are **bolded** and highlighted in red are implied attributes for which a value is required when I is chosen for classtype

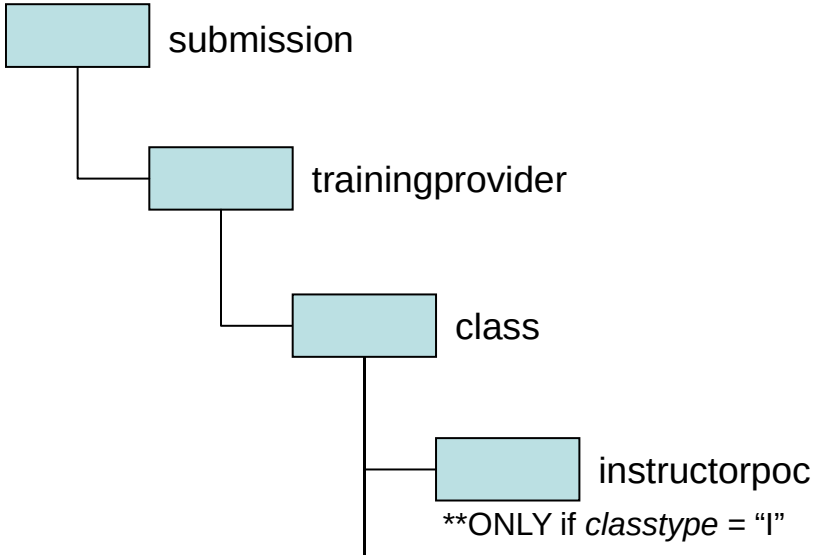
FEMA

Additionally, the element instructorpoc will also be required when I (Instructor-led) is chosen for classtype

The **instructorpoc** element is the child element of the **class** element

The element **instructorpoc** provides information on the class lead instructor

- The **instructorpoc** element is required ONLY if the *classtype* is "I"
- It contains four attributes
  - *instlastname*
  - *instfirstname*
  - *instphone*
  - *instemail*



Indicates that the element can be empty

```
<!ELEMENT instructorpoc EMPTY>
<!ATTLIST instructorpoc
          instlastname CDATA #REQUIRED
          instfirstname CDATA #REQUIRED
          instphone CDATA #REQUIRED
          instemail CDATA #IMPLIED>
```

FEMA

If the attribute *classtype* is "WD" for Web-based Domestic, then the following implied attributes will become required

<!ATTLIST class

classtype = "WD"

catalognum (AWR-103|AWR-110-W|AWR-111-W|…) #REQUIRED

classcity CDATA #REQUIRED

classstate (AL|AK|AZ|AR|CA|CO|…) #IMPLIED

classzipcode CDATA #IMPLIED

classcountry (AA|AC|AE|AF|…..) #IMPLIED

startdate CDATA #REQUIRED

enddate CDATA #REQUIRED

starttime CDATA #IMPLIED

endtime CDATA #IMPLIED

numstudent CDATA #REQUIRED

trainingmethod (R|M|I|W) #REQUIRED

contacthours CDATA #IMPLIED

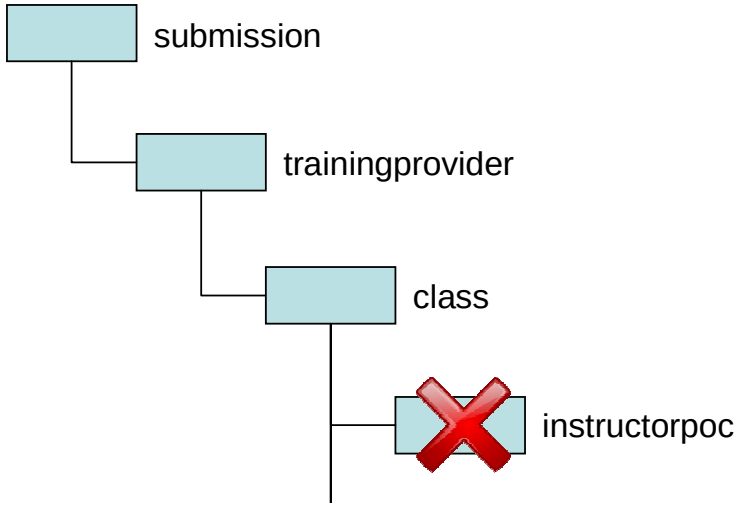
preparerlastname CDATA #IMPLIED

preparerfirstname CDATA #IMPLIED

batchpreparerphone CDATA #IMPLIED

batchprepareremail CDATA #IMPLIED>

Attributes that are **bolded** and highlighted in red are implied attributes for which a value is required



FEMA

If the attribute *classtype* is "WI" for Web-based International, then the following implied attributes will become required

<!ATTLIST class

classtype = "WI"

catalognum (AWR-103|AWR-110-W|AWR-111-W|…) #REQUIRED

classcity CDATA #REQUIRED

classstate (AL|AK|AZ|AR|CA|CO|…) #IMPLIED

classzipcode CDATA #IMPLIED

classcountry (AA|AC|AE|AF|…..) #IMPLIED

startdate CDATA #REQUIRED

enddate CDATA #REQUIRED

starttime CDATA #IMPLIED

endtime CDATA #IMPLIED

numstudent CDATA #REQUIRED

trainingmethod (R|M|I|W) #REQUIRED

contacthours CDATA #IMPLIED

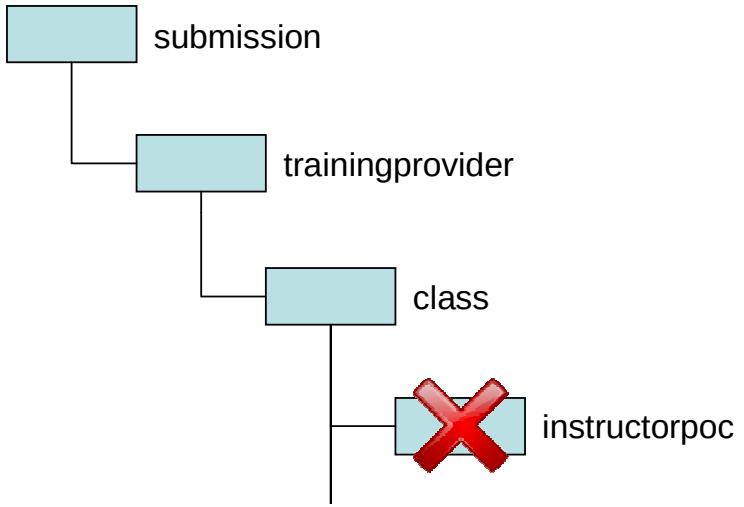
preparerlastname CDATA #IMPLIED

preparerfirstname CDATA #IMPLIED

batchpreparerphone CDATA #IMPLIED

batchprepareremail CDATA #IMPLIED>

Attributes that are **bolded** and highlighted in red are implied attributes for which a value is required



FEMA

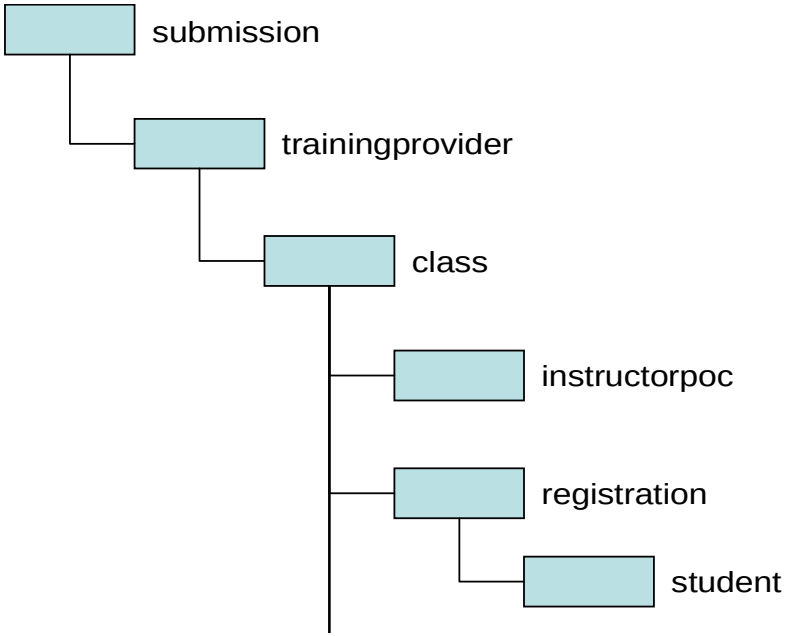
Element: registration

The **registration** element is the child element of the **class** element

The **registration** element does not have any attributes but has a child element, **student**

The **student** element provides general information of the student

- It has a combination of required and implied attributes
- Pay particular attention to the attribute, *international*



Indicates at a minimum one student information required

```
<!ELEMENT registration (student+)>
<!ELEMENT student EMPTY>
<!ATTLIST student international (Y|N) #REQUIRED
```

FEMA

If the attribute *international* is “N”, then the following implied attributes are required

<!ATTLIST student

International = “N”

studentlastname CDATA #REQUIRED

studentfirstname CDATA #REQUIRED

studentmi CDATA #IMPLIED

agency CDATA #IMPLIED

title CDATA #IMPLIED

address1 CDATA #IMPLIED

address2 CDATA #IMPLIED

address3 CDATA #IMPLIED

studentcity CDATA #REQUIRED

studentzipcode CDATA #IMPLIED

studentstate (AL|AK|AZ|AR|CA|…) #IMPLIED

studentcountry (AA|AC|AE|AF|AG|…) #IMPLIED

studentphone CDATA #REQUIRED

studentemail CDATA #IMPLIED

discipline (LE|EMS|EM|FS|…) #REQUIRED

govnlevel (L|S|DF|NF|NA) #REQUIRED>

Attributes that are **bolded** and highlighted in red are implied attributes for which a value is required

FEMA

If the attribute *international* is “Y”, then the following implied attributes are required

<!ATTLIST student

international = “Y”

studentlastname CDATA #REQUIRED

studentfirstname CDATA #REQUIRED

studentmi CDATA #IMPLIED

agency CDATA #IMPLIED

title CDATA #IMPLIED

address1 CDATA #IMPLIED

address2 CDATA #IMPLIED

address3 CDATA #IMPLIED

studentcity CDATA #REQUIRED

studentzipcode CDATA #IMPLIED

studentstate (AL|AK|AZ|AR|CA|…) #IMPLIED

studentcountry (AA|AC|AE|AF|AG|…) #IMPLIED

studentphone CDATA #REQUIRED

studentemail CDATA #IMPLIED

discipline (LE|EMS|EM|FS|…) #REQUIRED

govnlevel (L|S|DF|NF|NA) #REQUIRED>

Attributes that are **bolded** and highlighted in red are implied attributes for which a value is required

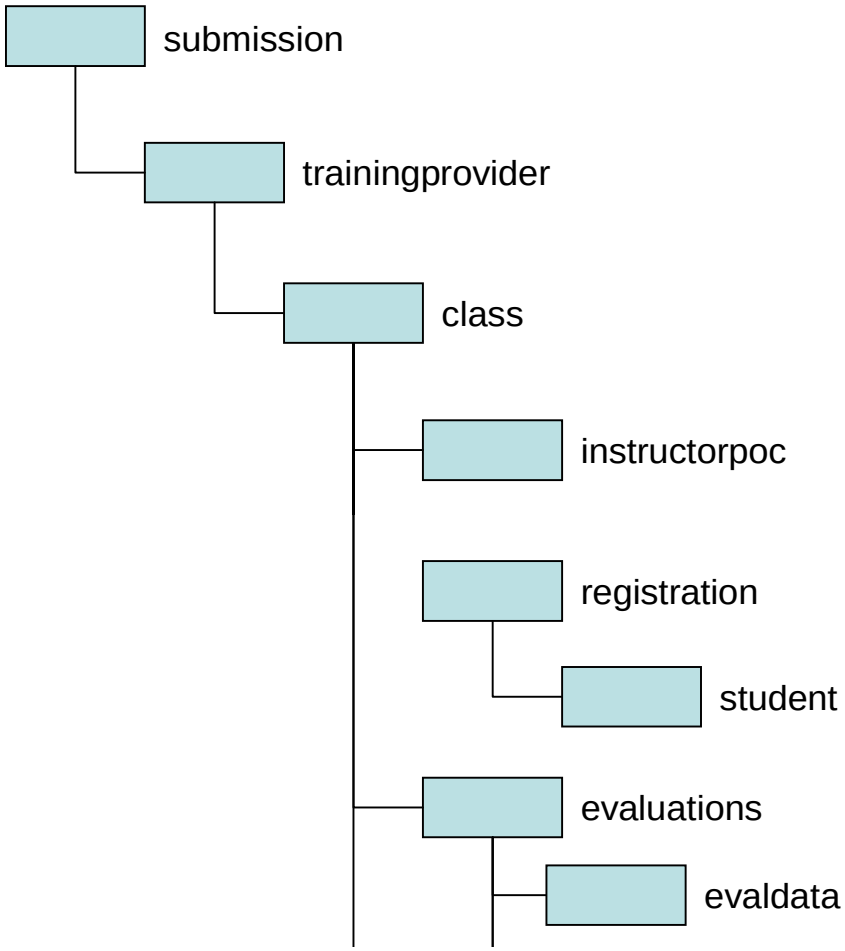
FEMA

Element: evaluations

The **evaluations** element is the child element of the **class** element

The **evaluations** element provides information on the student's ratings and evaluations of the course and instructor

The **evaluations** element does not have any attributes but has a child element, **evaldata**



FEMA

Element: evaldata

The **evaldata** element has two child elements

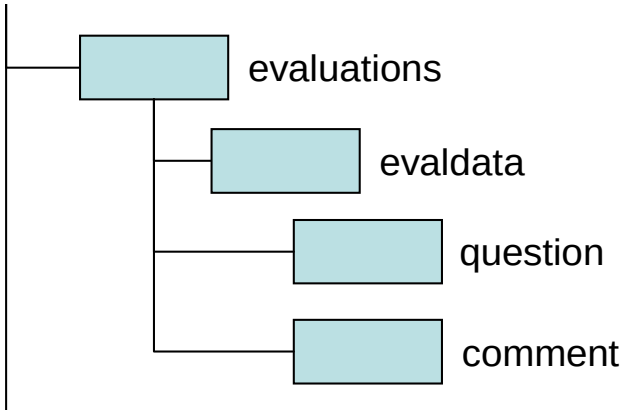
- **question**
- **comment**

The **question** element has two attributes

- *question id*
- *answer*

The **comment** element has two attributes

- *comment id*
- *answer*




```
<!ELEMENT evaluations (evaldata+)>
<!ELEMENT evaldata (question+, comment*)>
<!ELEMENT question EMPTY>
        <!ATTLIST question id (1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |…) #IMPLIED
        answer (0 | 1 | 2 | 3 | 4 | 5) #IMPLIED>
<!ELEMENT comment EMPTY>
        <!ATTLIST comment id (24 | 25 | 26 | 27) #IMPLIED
        answer CDATA #IMPLIED>
```

Indicates at a minimum one question provided

Indicates that comment may be one or zero

FEMA

Element: testaverage

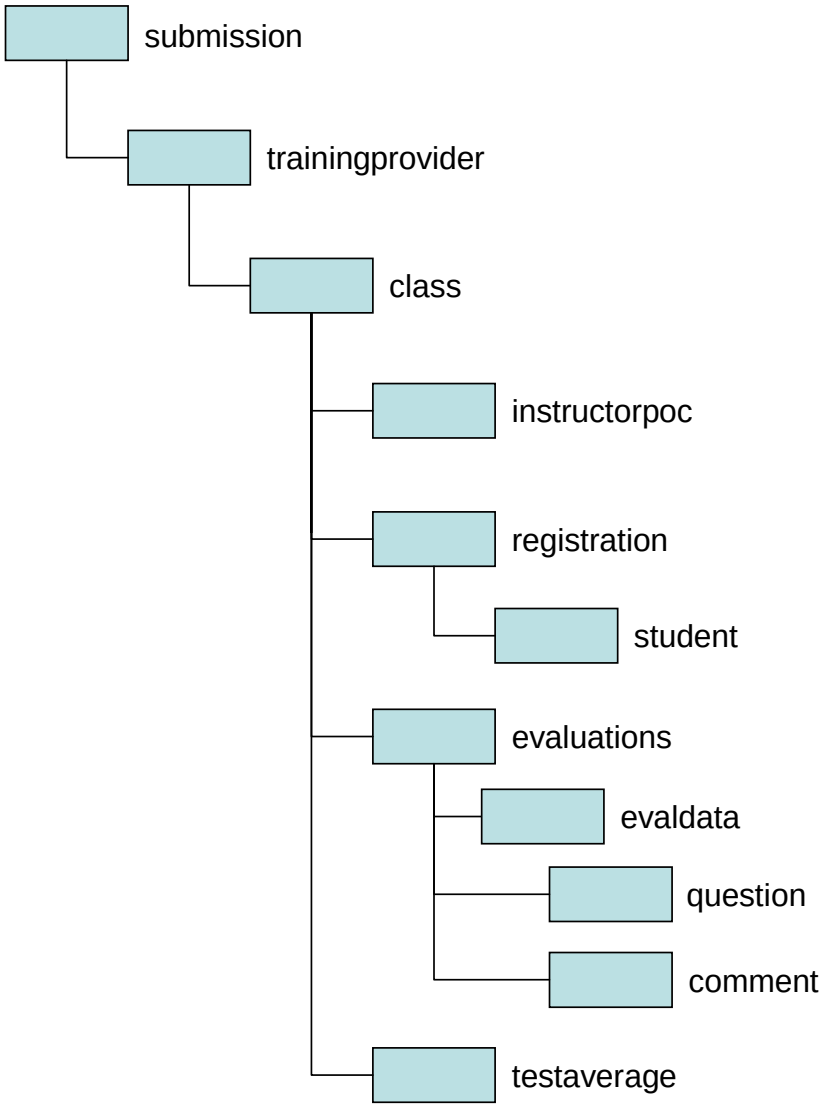
The **testaverage** element is the child element of the **class** element

The **testaverage** element has two attributes

- *pretest*
- *posttest*

Indicates that the element can be empty

```
<!ELEMENT testaverage EMPTY>
<!ATTLIST testaverage
          pretest CDATA #IMPLIED
          posttest CDATA #IMPLIED>
```



FEMA

XML File Naming Convention

The XML file shall be named in the following manner:

TP_CourseNumber_Date_SequenceNumber.XML

Where TP is the training provider acronym, CourseNumber is the catalog number of the course held, and Date (mmddyyyy) is the current date of submission. For XML files that have the same file name, the TP shall add a sequence number to indicate that there are XML files with the same name but different data within it.

Examples:

- MSU-CERT_AWR-189-1_06182008.XML
- MSU-CERT_AWR-189-1_06182008_1.XML

Submitting XML File to the RES

Submission Via the RES

- The TP can submit the XML file to the RES through the Submission Module.
- The system will provide an immediate notification for the TP to determine if the XML file was successfully or unsuccessfully transmitted to the queue for processing.
- Once the XML file has been successfully loaded into the RES, the submitter and the TP Point of Contact (POC) will receive a notification email

Submission Error Handling

- Once the XML file is in queue, it will undergo a validation process to ensure that the XML meets all of the required elements and attributes.
- If the RES determines that the XML does not meet the required elements and attributes, an email will be sent to the TP with a log summarizing the errors in the XML.

Resubmitting via the RES

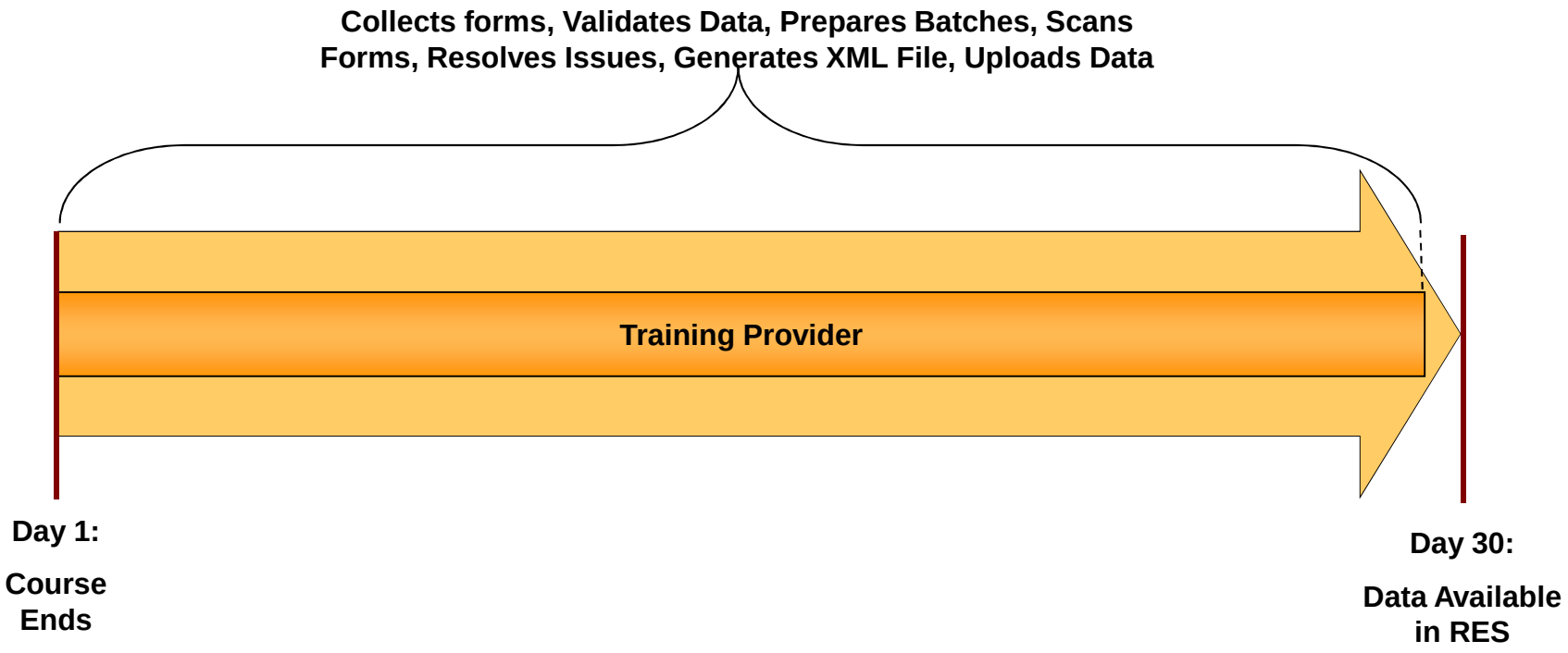
- Once the TP has addressed the errors in the XML, the TP can re-submit the XML. The filename of the XML must be different from the previously submitted file. Either the date or the sequence number must be changed. The XML will once again undergo the validation process.
- This cycle is repeated until the RES has determined that the XML has successfully passed the validation process and can be loaded into the RES database. The TP will receive an email indicating a successful load

FEMA

Timeline for XML Data Submission

TP will have 30 days after the course end date to submit their data

Data will be available in the RES immediately after upload



FEMA

Exercise 3: Writing an XML based on the RES DTD

Using the RES DTD "**submission.dtd,**" create an XML representing data from your Training Provider (TP) and submit it to the RES

The XML should consist of one instructor-led class with:

- Two students (one domestic, one international)
- At least one evaluation set
- At least one class pre and/or post test score(s)

Please use any dates for the month of May to represent the class start and end date

Don't forget about the file name convention!

FEMA

Training Session Summary

At the end of this training, each TP should be able to:

- ☑ Understand the concepts of an XML
- ☑ Write an XML
- ☑ Understand the concepts of a DTD and interpret its syntax
- ☑ Write an XML based on specified DTD
- ☑ Understand the basic concepts of data mapping
- ☑ Name an XML file based on the RES File Name Convention
- ☑ Write an XML based on specified RES DTD
- ☑ Submit an XML file via the RES
- ☑ Interpret an error log (if applicable)
- ☑ Re-name an XML file and re-submit via the RES (if applicable)

FEMA

User Support Contact Information

RES Help Desk and Support

– Email res@dhs.gov

FEMA

FEMA